A Mean-based Approach for Real-Time Planning

Damien Pellier, Bruno Bouzy, Marc Métivier

A Mean-based Approach for Real-Time Planning

(Extended Abstract)

Damien Pellier, Bruno Bouzy, Marc Métivier
Laboratoire d'Informatique de Paris Descartes
Université Paris Descartes
{damien.pellier,bruno.bouzy,marc.metivier}@parisdescartes.fr

ABSTRACT

In this paper, we introduce a new heuristic search algorithm based on mean values for real-time planning, called MHSP. It consists in associating the principles of UCT, a bandit-based algorithm which gave very good results in computer games, and especially in Computer Go, with heuristic search in order to obtain a real-time planner in the context of classical planning. MHSP is evaluated on different planning problems and compared to existing algorithms performing on-line search and learning. Besides, our results highlight the capacity of MHSP to return plans in a real-time manner which tend to an optimal plan over the time which is faster and of better quality compared to existing algorithms in the literature.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Heuristic methods, Plan execution, formation, and generation

General Terms

Algorithms, Experimentation

Keywords

Agent Reasoning, Planning, Learning

1. INTRODUCTION

The starting point of this work is to apply UCT [7], an efficient algorithm well-known in the machine learning and computer games communities, and originally designed for planning, on classical planning problems. UCT is designed for MDP, and based on bandit decision making [1]. In the background of the current paper that stresses time constraints, the interesting feature of UCT is its anytime property in a strong meaning. At any time, UCT is able to return the first action of a plan, a partial plan, a solution plan, or an optimal plan according to the given time. However, [7] did not give known successful applications in the classical planning domain yet. Instead, UCT gave tremendous results in computer games, and specifically in Computer Go with the Go playing program Mogo [4]. In Computer Go, UCT is efficient because the Go complexity is high, and because the Go games are played in real time, which fits the anytime property of UCT. Therefore, this paper focuses on how to give value to UCT-like algorithms in a sub-field of planning dealing with time constraints.

Planning can be divided into off-line and on-line planning. In off-line planning, the planners build solution plans, and then execute the all the actions of the plan. They use very good heuristic functions [2] that drive the search efficiently toward the goal [6]. However, although they find solution plans very quickly on sufficiently easy problems, these may fall if they have not enough time to find a solution plan.

Conversely, in on-line planning, the agent repeatedly executes a task, called an episode or a trial. At each step, the agent is situated in a current state, and performs a search within the space around the current state in order to select his best action. Then the agent executes the action, and reaches a new state. When the agent reaches the goal state, another trial or episode is launched, and so on. When the planners make their decision in constant time, this corresponds to Real-Time Search (RTS). Since the pioneering Korf's work on puzzles [9], RTS has been strongly linked, to the development of video games in which the agents need good path-finding algorithms running in real time. RTS was not broadened to the classical planning problems. There are several real-time searches, e.g., mini-min [8], γ-Trap and LRTS [3], or even A* [5]. These algorithms perform action selection by using heuristic search. The Korf's work shows the importance of an efficient search for action selection. Consequently, it is of interest to see if an adaptation of UCT can be used efficiently for action selection in RTS.

The aim of this paper is to focus on the action selection stage of RTS, to present MHSP, an heuristic search algorithm based on mean values, i.e., an adaptation of UCT to RTS, adapted in the context of classical planning. We show that MHSP performs better for action selection in the RTS background, with or without learning, than the existing real-time action selectors.

2. UCT

UCT worked well in Go playing programs, and it was used under many versions. However, UCT basically works as follows. While time remains, UCT iteratively grows up a tree in the computer memory by following the steps below:

1. Starting from the root, browse the tree until reaching a leaf by using (1).
2. Expand the leaf with its child nodes.

Cite as: Mean-based Heuristic Search for Real-Time Planning (Extended Abstract), Damien Pellier, Bruno Bouzy and Marc Métivier, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX.

3. Choose one child node.

4. Perform a random simulation starting from this child node until the end of the game, and get the return, i.e., the game's outcome.

5. Update the mean value of the browsed nodes with this return.

With infinite time, the root value converges to the minimax value of the game tree [7]. The UCB selection rule (1) answers the requirement of being optimistic when a decision must be made facing uncertainty [1].

$$N_{select} = \arg\max_{n \in N} \{m + C\sqrt{\frac{\log p}{s}}\} \quad (1)$$

N_{select} is the selected node, N is the set of children, m is the mean value of node n, s is the number of iterations going through n, p is the number of iterations going through the parent of n, and C is a constant value setup experimentally. Equation (1) uses the sum of two terms: the mean value m, and the UCB bias value which guarantees exploration. The respect of the UCB selection rule guarantees the completeness and the correctness of the algorithm.

3. MHSP

This section highlights the two important choices made in designing MHSP compared to UCT.

3.1 Heuristic values replace simulation

On planning problems, browsing randomly the state space from leaf nodes of UCT tree does not enable the algorithm to reach goal states sufficiently often. Many runs complete without reaching goal states. Therefore, replacing the simulations by a call to the heuristic becomes relevant.

In Computer Go, the random simulations were adequate mainly because they always completed after a limited number of moves, and the return values (won or lost) were roughly equally distributed on most positions of a game. In planning, no such garantee exists. Furthermore, the two return values correspond to actual values of a completed game. In planning, one return means that a solution has been found (episode completed), and the other not. This simulation difference is fundamental between the planning problem, and the two-player game playing problem.

In planning, the heuristic values bring appropriate knowledge into the returns. Consequently, using heuristic values in MHSP should be positive bound to the condition that the heuristic value generator is good, which is the case in planning [2]. In Computer Go, replacing the simulations by an evaluation function call is forbidden by fifty years of computer Go history which experienced the converse path: the complex evaluation functions have been replaced by pseudo-random simulations.

3.2 Optimistic initial mean values

Computer games practice shows that the UCB bias of (1) can merely be removed, provided the mean values of nodes are initialized with sufficiently optimistic values. This simplification removes the problem of tuning C. Generally, to estimate a given node, the planning heuristics give a path length estimation. Convergence to the best plan is provided by admissible heuristics, i.e., optimistic heuristics. Consequently, the value returned by planning heuristics on a node can be used to initialize the mean value of this node.

In MHSP, the returns are negative or zero, and they are in the opposite of the distance from s to g. With this initialization policy, the best node according to the heuristic value will be explored first. Its value will be lowered after some iterations whatever its goodness, and then the other nodes will be explored in the order given by the heuristic.

4. CONCLUSION

In this paper, we presented a new heuristic search algorithm MHSP, based on mean values for RTS, and adapted in the context of classical planning. This algorithm combines heuristic search and principles of UCT: state values based on mean returns, and optimism in front of uncertainty.

MHSP computes mean values for decision and not direct values. It means that the value of an action depends on every nodes explored beneath that action, and not only on the best node found. This fact may have a strong impact on the way the system explores nodes because the system may focus on action permitting to reach globally good node, and not on the action enabling to reach the node with the best heuristic value. In a time constrained context, focusing on action which leads to globally good nodes instead of just one node may limit the effect of strongly erroneous heuristic values. It enables to subtly explore the tree. The more complex the problem is, the more visible should be this effect.

5. ACKNOWLEDGMENTS

This work has been supported by French National Research Agency (ANR) through COSINUS program (project EXPLO-RA number ANR-08-COSI-004).

6. REFERENCES

[1] P. Auer, N. Cesa-Bianchi, and P. Fisher. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47(2–3):235–256, 2002.

[2] A. Blum and M. Furst. Fast Planning Through Planning Graph Analysis. *Artificial Intelligence*, 90:1636–1642, 1997.

[3] V. Bulitko and G. Lee. Learning in Real-Time Search: A Unifying Framework. *Journal of Artificial Intelligence Research*, 25:119–157, 2006.

[4] S. Gelly, Y. Wang, R. Munos, and O. Teytaud. Modification of UCT with Patterns in Monte-Carlo Go. Technical Report RR-6062, INRIA, 2006.

[5] P. Hart, N. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics SSC4*, 2:100—107, 1968.

[6] J. Hoffmann and B. Nebel. The FF Planning System: Fast Plan Generation Through Heuristic Search. *JAIR*, 14(1):253–302, 2001.

[7] L. Kocsis and C. Szepesvari. Bandit-based Monte-Carlo Planning. In *Proc. ECML*, pages 282–293, 2006.

[8] R. Korf. Real-Time Heuristic Search: New Results. In *Proceedings of the AAAI conference*, pages 139–144, 1988.

[9] R. Korf. Real-Time Heuristic Search. *Artificial Intelligence*, 42(2-3):189–211, 1990.